OurBook+

Software Architecture Document

Version <1.0>

Revision History

| **Date** | **Version** | **Description** | **Author** |
| --- | --- | --- | --- |
| 09/Dec/2022 | 1.0 | **- Kỳ:**  + Architecture goal and constraints  + Draw software architecture diagram and describe  + Component of front end  + Class diagram: book, printed book, e-book, search book UI, view book detail UI, view book detail ebook UI, view book detail printed book UI, view book detail any format UI, borrow book UI  + Explain classes  - **Uyên:**  + Architecture goal and constraints  + Component of front end  + Class diagram: book, printed book, e-book, Q&A UI, cart, view cart UI, share book UI, share ebook UI, share printed book UI, login UI, signup UI  + Explain classes  **- Hoàng (not complete):**  + Component of backend and database  + Class diagram: appointment, view appointment, login, signup  + Explain classes  **- Minh:**  + Component of backend and database  + Class diagram: user, book, admin, book borrower controller  + Explain classes | Đinh Mỹ Kỳ  Nguyễn Minh Uyên  Trương Nhật Minh |

Table of Contents

[**1. Introduction**](#_heading=h.gjdgxs) **4**

[**2. Architectural Goals and Constraints**](#_heading=h.30j0zll) **4**

[**3. Use-Case Model**](#_heading=h.1fob9te) **5**

[**4. Logical View**](#_heading=h.2et92p0) **6**

[**5. Deployment**](#_heading=h.3dy6vkm) **13**

[**6. Implementation View**](#_heading=h.1t3h5sf) **13**

Software Architecture Document

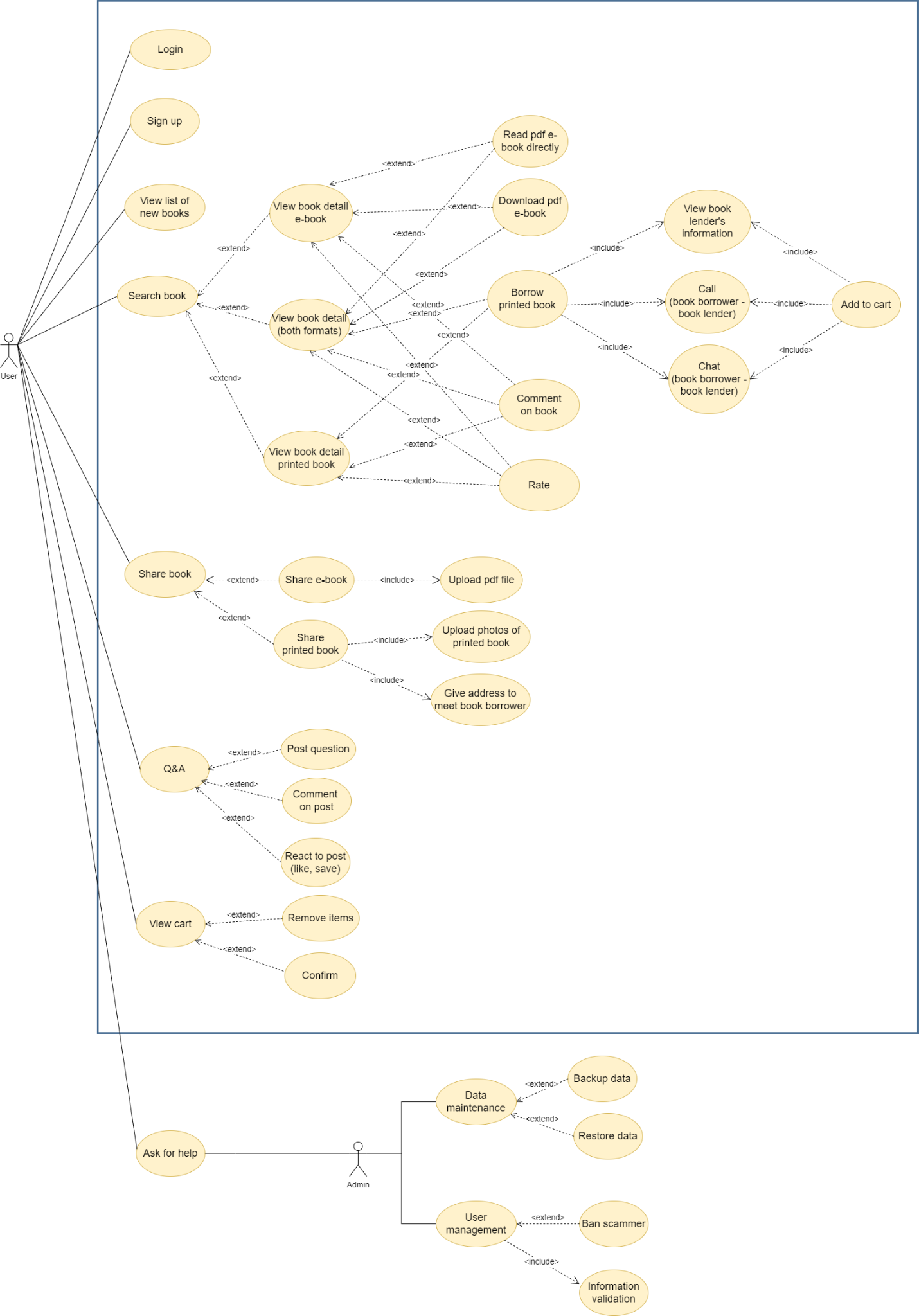
# Introduction

The introduction of the Software Architecture Document provides an overview of the entire Software Architecture Document. It includes the purpose, scope, definitions, acronyms, abbreviations, references, and overview of the Software Architecture Document.

# Architectural Goals and Constraints

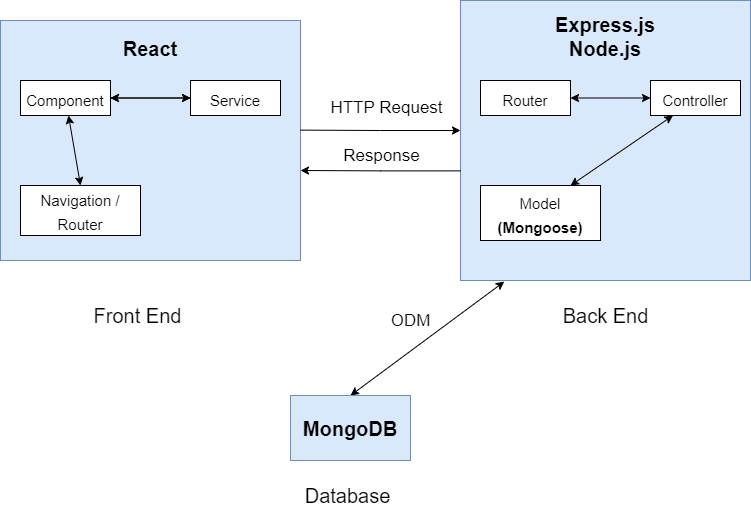
* Goal:
* Safety, privacy: User’s information is protected and safe
* Security:
* Login: Use a username and a password
* The system must be secured for users to read ebook online, make personal appointments to borrow books, and protect users’ personal information when they share addresses to meet other users and lend books.
* Persistence: ensure non-functional requirements as described in the vision document.
* User friendly: Requests from users are processed without errors and processed fastly. Thus, components are tight enough but not too tight that affect maintainability.
* Maintainability: Components are self-contained, no need to fix several different components when a problem occurs
* Portability: Easily accessible by different devices
* Constraints
* Development tools: MongoDB, ExpressJs, React, NodeJs (MERN Stack)
* Implementation: OurBook+ is implemented as a client-server web application with admin and users. Admin manages user’s information and assists users if they need help.

# Use-Case Model



# Logical View

* OurBook+ software architecture diagram:



+ Front End sends HTTP requests to Back End. The Back End receives these requests and responses to the Front End.

+ API is called in Node.js and Express.js. We use Google Map API, and API for chat, call functions.

+ Mongoose used for model to connect Node,js with MongoDB

+ Use ODM (Object document mapper) to connect back end with MongoDB database and vice versa.

* Diagram for component of Front End:

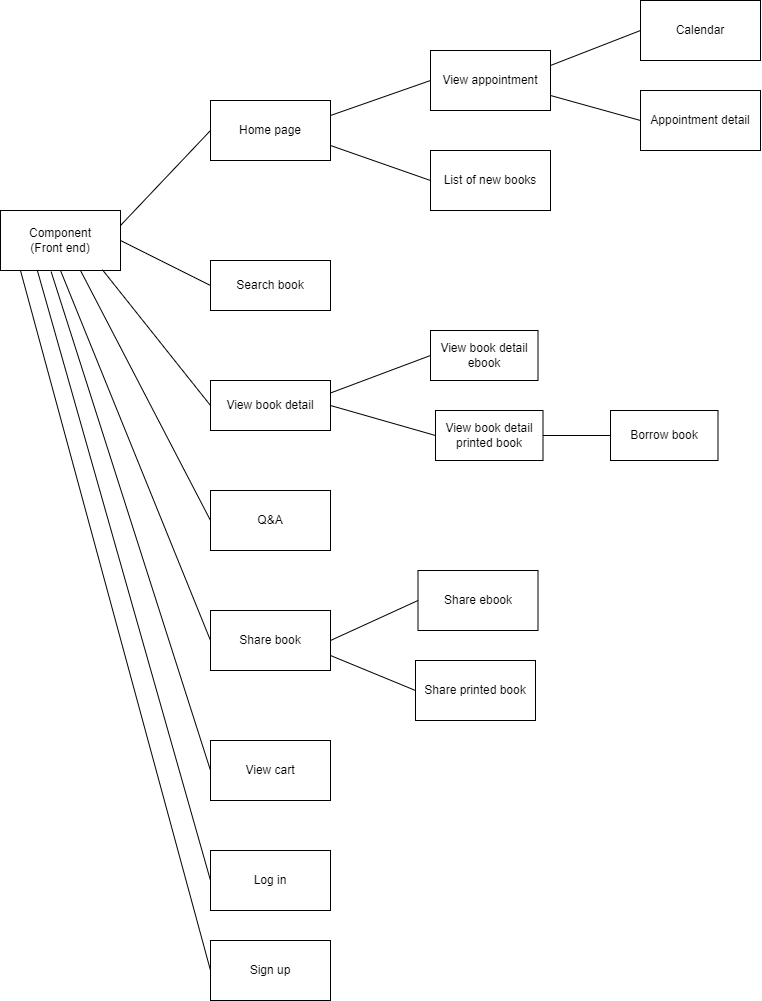


Diagram for server - client

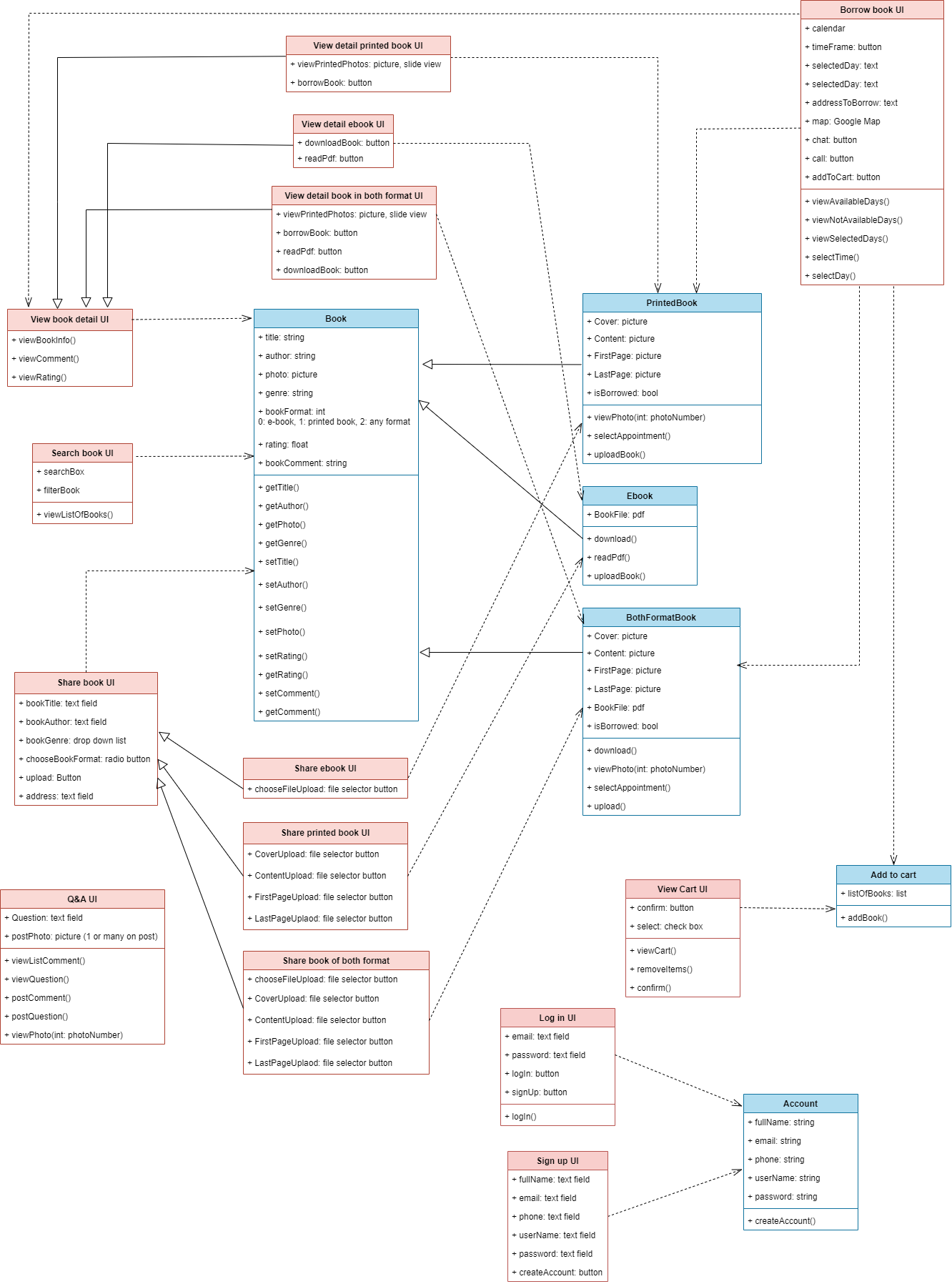
**

**4.1 Component of Front End:**

This component contains important classes for both front end and back end, as well as some classes for UI front end.

Note: The blue ones are main object classes, all the remaining classes are UI classes.

Class Diagram:



**Describe key classes**

1. *Class Book:*

* This class is used to store book information and its ratings, comments
* 1 of the most important classes for other UI classes to call.
* Book has setter, which can be used by user to upload new books, or admin to update book information
* getter: get book’s information. Other UI classes will use these getter functions to view book information to the UI.
* There are 3 types of book: printed book, ebook and book in both formats. Thus, here we use inheritance for these types as described below.

1. *Class Ebook:*

* This class is use to store ebook information
* Ebook file is pdf file
* User can download pdf or read pdf directly on website with different styles: page-by-page, 2 pages at the same time with flipping, zoom in/out
* Ebook is called by share ebook and view ebook detail

1. *Class PrintedBook:*

* This class is used to store printed book information.
* It is a derived class of class book which store printed book information
* In additional to book, it also stores 4 pictures taken from the printed books along with information whether it is borrowed or not
* User can view the photos and select appointment of a PrintedBook object and lender can upload a PrintedBook object
* PrintedBook is used by both share printed book and view printed book detail

1. *Class BothFormatBook:*

* This class is used to store the book which is in both format: ebook and printed book
* It is a derived class of class book which store book in both format information
* In additional to book, it also have ebook file which is pdf file, 4 pictures taken from the book which in printed form along with information whether it is borrowed or not
* User can download and read with different styles as in Ebook and can also view photos and select appointment as in PrintedBook
* Lender can upload a BothFormatBook object by providing all its attribute information.
* BothFormatBook is used by both share both format book and view detail book in both format

1. *Class Search book UI:*

* The class describes UI for searching book with search box to type in keyword; and filter with book genres, book formats
* Search book return list of books in results if the book is found in database, otherwise return not found
* Use book information to view a list of books.

1. *Class View book detail UI:*

* This is used to view book information, comments and ratings after user clicks on each book of search results.
* The class contains main parts of view book detail. There are 3 types of view book detail based on book formats: ebook, printed book or book in both formats, so we use inheritance for view book detail ebook, printed book and book in both formats.

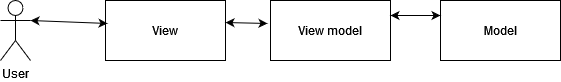
1. *Class Borrow book UI:*

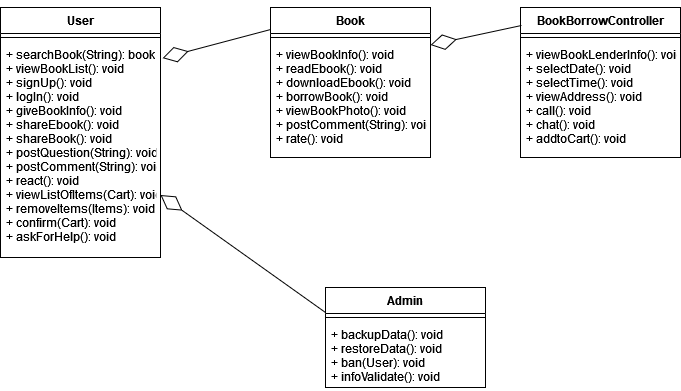
* This class describes how user view and borrow printed book
* The user can view the book’s details again (to make sure that the user is borrowing that book) before they can view the book lender’s information, make an appointment to borrow and add to cart => call view book detail
* Borrow book UI calls class printed book to know if the book is currently borrowed or not.
* The user can view available, not available days to borrow books, time frame to borrow. After the user chooses the day and time, it will also show in text.
* The user can see the address to meet the book lender to borrow the book (in text and Map), call and chat with the book lender.
* Finally, user must click on add to cart to add the printed book to cart => call class add to cart

1. *Class Share book UI:*

* *This class is used by the lender to upload(create) and store a new Book object to the system (Ebook or PrintedBook or BothFormatBook), the lender is required to give basic information for Book attributes.*
* *It has derived classes: Share ebook UI, Share printed book UI, and Share book of both format UI. Depending on the choice of type of book to upload, the lender need to add information base on the additional attributes of the class Ebook or PrintedBook or BothFormatBook*

**4.2 Component: Client**

**

**

**4.3 Component: Server**

****

# Deployment

# Implementation View